댁내 장치의 원격 제어를 위한 UPnP 프록시 시스템

(A UPnP Proxy System for the Remote Control of Home Appliances)

김 동 희 † 임 경 식 †† 이 화 영 †††
(Dong Hee Kim) (Kyungshik Lim) (Hwa Young Lee)

안 준 철 †††† 조 충 래 † 박 광 로 †††††
(Joon Chul Ahn) (Chunglae Cho) (KwangRoh Park)

요 약 본 논문에서는 홈 네트워크가 사설망으로 구성됨으로 인하여 인터넷과 홈 네트워크가 단절된 환경에서 사용자가 인터넷을 통하여 홈 네트워크 내부의 장치들을 제어하고 모니터링을 할 수 있는 UPnP 프록시 시스템을 제안한다. 홈 네트워크가 사설망으로 구성되는 환경에서는 댁내 장치의 광고 메시지가 인터넷으로 전송되지 않아 사용자는 댁내에 어떠한 장치가 접속되어 있는지 알 수 없으며, 사용자는 댁내의 장치에 직접 접근할 수 없기 때문에 장치 제어 명령을 해당 장치로 전달할 수 없다. 따라서 본 논문에서는 이러한 문제점을 분석하고 외부 인터넷에서 댁내 UPnP 장치들을 제어할 수 있는 방안을 제시한 후, UPnP 프록시 시스템을 설계하고 구현한다. 본 시스템의 장점으로, 사용자는 본 시스템을 사용하여 유무선 환경의 인터넷에서 기존의 댁내 환경에서의 장치 제어를 위한 프리젠테이션 문서를 이용하여 장치를 제어할 수 있으며, HTTP 기반의 푸시 기술을 통하여 사용자는 홈 네트워크의 상태를 실시간으로 모니터링 할 수 있다.

키워드 : 홈 네트워크, 미들웨어, UPnP, UPnP 프록시

Abstract Because of a security problem and not enough IPv4 address space, the home network has been made up of private network, and it has been separated from Internet. This fact prevents people in Internet from controlling and monitoring home appliances. So, this paper designs and implements the UPnP Proxy System which offers functions for users to control and monitor home appliances. When users are in the outside of the home network, they do not know which devices were connected in the home network because the advertisement messages of UPnP devices would not be delivered to the outside of the home network. Also, users cannot access devices directly, and their control messages are not delivered into the home network. So, this paper designs and implements the UPnP Proxy System to solve these problems. The merit of the system is that users can control and monitor home appliances in realtime using presentation web documents with the HTTP push technology.

Key words : home network, middleware, UPnP, UPnP Proxy

† 비 회 원 : 한국전자통신연구원 홈네트워크그룹 연구원
donghee@etri.re.kr
clcho@etri.re.kr
†† 종신회원 : 경북대학교 컴퓨터과학과 교수
kslim@knu.ac.kr
††† 비 회 원 : LG전자 솔루션기술그룹 연구원
hylee04@lge.com
†††† 비 회 원 : 유시스 디바이스지원팀
jcahn@usys.co.kr
††††† 비 회 원 : 한국전자통신연구원 디지털홈연구단 그룹장
krpark@etri.re.kr
논문접수 : 2003년 9월 23일
심사완료 : 2004년 5월 18일

1. 서 론

홈 네트워크는 다양한 분야의 기술들이 모여 이루어진다. 하부의 물리적 통신 장비부터 응용 프로그램까지 폭넓은 기술의 집합체이다. 이 중, 미들웨어 기술은 상위의 응용 프로그램에게 하부 물리적 장치의 일반적인 제어 방법을 제공한다. 현재 Universal Plug and Play (UPnP), Jini, Open Service Gateway Initiative (OSGi), LonWork, Havi 등 다양한 미들웨어 기술들이

제안되어 있으며, 이들 중 UPnP와 Jini는 기존의 인터넷 기술을 기반으로 하는 기술이다. 물론 예전부터 홈오토메이션, 홈 시큐리티 등의 홈 네트워크와 유사한 서비스가 존재하였으나, 이들은 고립되고 제한적인 서비스밖에 제공할 수 없었다. 그러나 인터넷 기반의 홈 네트워크 기술은 인터넷의 대중화를 바탕으로 보편적이고 폭넓은 서비스를 제공할 수 있다.

그러나 UPnP[1,2]와 Jini[3]가 인터넷 기술을 이용한 미들웨어라고 할지라도 사용자가 집 밖에서 인터넷을 통하여 댁내의 가전 기기들을 제어하고자 할 때에는 이들 기술을 그대로 사용할 수 없다. 이는 이 기술들이 가지는 기본적인 동작 방법 때문이며, 이를 위해서는 이들 미들웨어의 기능들을 확장할 필요가 있다. UPnP와 Jini는 동일한 네트워크 상에서 장치를 제어하기 위한 구조와 메시지의 형태 등을 정의하고 있으나 인터넷과 홈 네트워크가 분리되는 환경에서는 인터넷의 사용자가 홈 네트워크 내부의 장치들을 제어할 수 없다. 따라서 본 논문은 기존의 미들웨어 기술을 확장하여 이들 제어 미들웨어들을 실제 홈 네트워크 환경에 적용하고, 사용자는 언제 어디서나 홈 네트워크의 장치들을 모니터링하고 필요할 경우 장치들을 제어할 수 있는 시스템을 설계하고 구현한다.

UPnP 스펙은 사용자의 제어 인터페이스인 HTML 프리젠테이션 문서를 옵션 기능으로 정의하고 있으며 현재 많은 응용의 사용자 인터페이스는 WEB 기술을 기반으로 개발되고 있는 추세로 인하여 대부분의 장치들은 프리젠테이션 문서를 통하여 장치 제어 인터페이스를 제공할 것으로 예상된다. 그러나 이러한 프리젠테이션 문서와 기존의 컨트롤 포인트 모듈은 앞서 언급한 인터넷과 홈 네트워크가 분리된 환경에서는 정상적인 동작을 하지 않는다. 본 시스템은 이러한 문제점을 해결하여 사용자에게 집 밖에서도 집 안과 동일한 제어 환경을 제공하기 위하여 개발되었다. 즉, 유무선 인터넷에서도 사용자가 프리젠테이션을 사용하여 댁내 장치들을 발견하고 제어하며, 실시간으로 장치의 현재 상태를 검사하기 위하여 UPnP 프록시 시스템을 개발하였다.

본 논문에서는 기존의 장치 제어 미들웨어의 한계는 무엇이며 본 연구의 개발물인 UPnP 프록시 시스템이 이를 어떻게 극복하였는지에 대하여 기술하고자 한다. 2장에서는 외부 인터넷에서 홈 네트워크의 장치를 제어할 때 Jini와 UPnP의 한계점과 UPnP를 사용하였을 때 이 문제를 해결할 수 있는 방법에 대하여 기술한다. 3장에서는 본 시스템의 구조와 동작 방법에 대하여 설명하고 본 시스템의 결과를 보인다. 그리고 마지막으로 4장에서 결론을 맺도록 한다.

2. UPnP 프록시 시스템의 요구사항 분석 및 사례 연구

홈 네트워크에는 여러 종류의 가전기기들이 서로 연결되어 사용자에게 다양한 서비스를 제공한다. 사용자는 댁내에서 네트워크에 연결된 장치들을 수시로 모니터링하고 제어할 수 있으며, 중요한 이벤트가 발생하였을 경우 장치는 이를 사용자에게 능동적으로 알려줄 수 있다. 또한, 서로 다른 장치들이 연동하여 사용자에게 새로운 서비스를 제공할 수 있다. 그러나 실제적인 홈 네트워크 서비스를 제공하기 위하여 이러한 서비스들은 사용자가 댁내에 있을 때뿐만 아니라 집 밖에 있을 때에도 이러한 서비스를 제공 받을 수 있어야 한다.

홈 네트워크는 현재 IPv4 주소의 부족과 보안 문제로 인하여 사설망으로 주로 구성되며, 홈 네트워크와 인터넷은 기본적으로 서로 단절된 별개의 네트워크이다. IPv6가 활발히 표준화가 진행되고 있으나 아직은 해결되지 못한 기술적 문제점, IPv4와의 호환성, 그리고 보급화 문제 등으로 홈 네트워크를 IPv6로 구성하기에는 아직 문제가 있다. 그리고 홈 네트워크 내의 기기들을 일관된 방법으로 제어하기 위해 제안된 Jini와 UPnP 등의 제어 미들웨어들은 홈 네트워크 내부에서 장치를 제어할 때에는 정상적인 동작을 하지만, 위와 같은 환경에서 이들을 집 밖의 인터넷을 통하여 제어하기에는 각 기술에 따라 다양한 문제점이 발생한다.

2.1 Jini[3] vs. UPnP[1,2]

Jini는 Sun Microsystems를 중심으로 한 Jini 포럼에서 제안하였으며, Java 언어를 기반으로 한다. Jini는 사용자에게 서비스를 제공하는 장치와 이들을 등록하는 룩 업 서비스로 구성되어 있다. 장치들은 자신이 제공하는 서비스 프록시를 룩 업 서비스에 등록하며, 사용자는 자신이 원하는 서비스를 룩 업 서비스에서 검색하고 장치와 통신할 수 있는 서비스 프록시 모듈을 다운로드 함으로써 자신이 원하는 서비스를 제공받을 수 있다.

그러나 서비스 프록시를 다운로드받아 장치를 제어하는 방법은 인터넷을 통하여 Jini 기반의 홈 네트워크 장치들을 제어하고자 할 때 문제점을 발생한다. 서비스 프

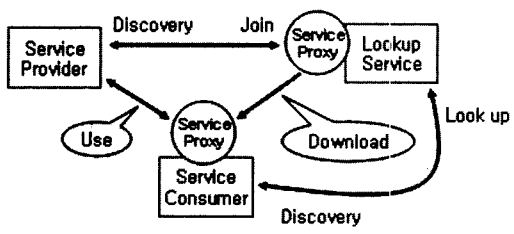

그림 1 Jini의 서비스 제공 구조

록시는 Java로 구현된 인터페이스로 바이너리 코드이다. 클라이언트로 전송된 프록시는 홈 네트워크 내부의 장치로 접속을 시도하지만 그림 2의 (a)처럼 사설 IP 주소를 가지는 장치로 인터넷에서는 직접 접속할 수 없다. 댁내 장치로의 직접적인 접속을 위해서는 NAT(Network Address Translation), 그리고 포트 매핑 등의 기능을 이용할 수 있지만, 이는 장치의 개수가 많아지면 관리가 어려워지며 장치가 외부에 바로 노출이 되므로 보안상 문제가 될 수 있다. 그리고 Jini는 홈 네트워크의 서비스를 제공하기 위한 프레임워크를 정의하고 있지만 클라이언트와 장치 간의 메시지의 형태는 정의하지 않는다. 따라서 클라이언트와 장치가 그림 2의 (b)처럼 홈 게이트웨이의 메시지 전달 모듈을 통하여 접속이 이루어지더라도 Jini가 가정하는 클라이언트와 장치 간의 환경과 실제의 환경이 다르기 때문에 정상적인 장치의 동작을 보장할 수 없다. 또한, 앞서 언급하였듯이 서비스 프록시와 장치 간의 메시지는 응용에 따라 서로 다르게 정의되기 때문에 홈 게이트웨이에 메시지 전달 모듈을 두기에도 상당한 어려움이 있다.

만약 서비스 프록시를 이용하지 않고 장치를 제어하고자 한다면, 홈 게이트웨이는 서비스 프록시와 장치에 대한 디스크립션을 기반으로 제어 인터페이스를 시스템이 동적으로 생성하여 사용자에게 제공하여야 한다. 그러나 해당 장치에 대한 충분한 사전 정보가 없다면 해당 장치에 최적화된 제어 인터페이스를 생성하는 것은 불가능하다.

UPnP는 Microsoft를 중심으로 한 UPnP 포럼에서 제안한 제어 미들웨어로서 HTTP, Simple Service Discovery Protocol(SSDP), Simple Object Access Protocol(SOAP), General Event Notification Architecture(GENA) 등 인터넷에서 사용되는 기존의 프로토콜을 사용한다. 주요 구성요소는 UPnP 장치와 컨트롤 포인트이며, 컨트롤 포인트는 SSDP와 GENA를 사용하여 홈 네트워크에 연결된 장치를 발견한다. 장치는 자신의 서비스를 기술하는 장치 디스크립션과 서비스 디스크립션을 가지고 있어 이를 컨트롤 포인트에게 전송하며 컨트롤 포인트는 이를 기반으로 장치를 제어한다. 그러나 이 기술 또한 Jini처럼 인터넷을 통하여 UPnP 기반의 홈 네트워크 장치들을 제어하고자 할 때에는 문제점을 발생한다.

UPnP를 서버와 클라이언트 모델로 보았을 때, 장치는 서비스를 제공하는 서버이며 컨트롤 포인트는 클라이언트이다. 그리고 사용자는 컨트롤 포인트를 제어하여 자신이 원하는 서비스를 제공받는다. 그러므로 컨트롤 포인트는 사용자의 컴퓨터에 존재하여야 한다. 그러나 사용자가 홈 네트워크 외부에 존재할 경우 그림 3의 (a)처럼 컨트롤 포인트는 장치의 광고 메시지를 전송받을 수 없다. 이러한 문제는 홈 네트워크가 사설망으로 구성되기 때문에 발생한다. 또한 홈 네트워크가 사설망으로 구성되지 않더라도 스펙에서 정의된 광고 메시지의 전달 범위의 한계로 인하여 이러한 문제가 발생한다.

그러나 이러한 문제점은 본 시스템이 제안한 방법인

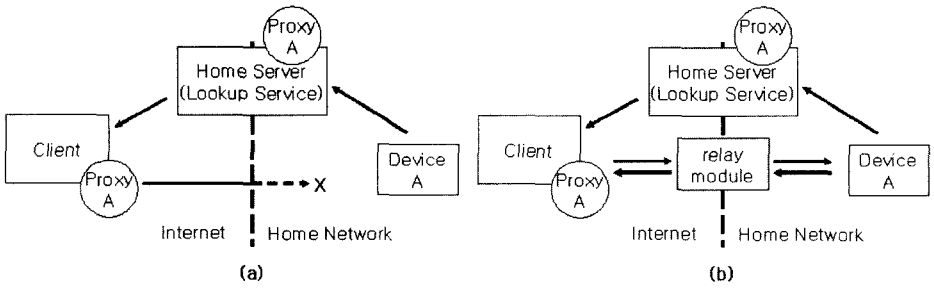

그림 2 Jini를 이용한 장치제어의 문제점

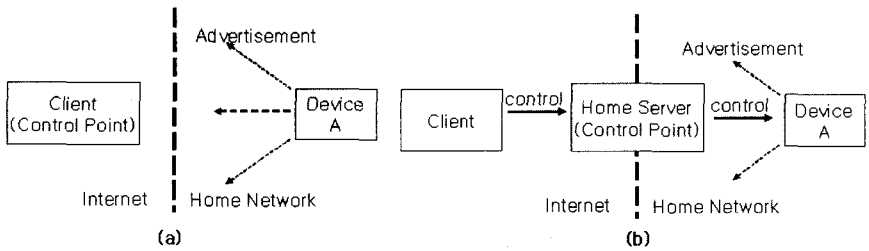

그림 3 UPnP를 이용한 장치제어의 문제점과 그 해결책

그림 3의 (b)와 같이 홈 네트워크와 인터넷이 연결되는 홈 게이트웨이에 컨트롤 포인트를 두어 이러한 문제점을 해결할 수 있다. 컨트롤 포인트는 장치의 광고 메시지를 수신하여 장치의 존재 여부를 알 수 있으며 이를 사용자에게 어떠한 형태로든지 알려줄 수 있다. 또한 사용자는 원격으로 컨트롤 포인트를 제어하여 결과적으로 사용자는 인터넷에서 댁내의 장치를 제어할 수 있다. 이는 UPnP의 모든 메시지가 XML로 기술되어있기 때문에 컨트롤 포인트 기능을 가지는 홈 게이트웨이가 클라이언트와 장치 간의 메시지를 적절히 수정하여 전달함으로써 그림 2 (b)의 문제점을 해결할 수 있다.

본 논문에서 제안하는 UPnP 프록시 시스템은 UPnP 기술을 확장하였다. UPnP를 확장하는 것은 Jini를 확장하는 것 보다 사용자에게 제어 인터페이스를 제공함에 있어 장점을 가지기 때문이다. 이 두 미들웨어에서 사용자의 제어 인터페이스로 제공되는 것은 UPnP의 프리젠테이션 문서와 Jini의 서비스 프록시다. 그러나 앞서 언급하였듯이 Jini에서는 장치의 서비스 프록시를 홈 네트워크 외부의 클라이언트로 전송하여 장치를 제어하고자 할 때 그 메시지 형태와 전송 프로토콜이 응용에 따라 다르기 때문에 홈 게이트웨이는 클라이언트와 장치 간의 메시지를 중계할 수 없어 사용자는 서비스 프록시를 사용할 수 없다. 그러나 UPnP에서의 메시지는 텍스트 형태의 XML 문서이며 HTTP를 통하여 전송되기 때문에 그림 2의 (b)와 그림 3의 (b)처럼 홈 게이트웨이가 클라이언트와 장치 사이에서 메시지를 해석하고 필요한 작업을 수행할 수 있다. 따라서 사용자는 홈 네트워크 외부의 인터넷에서도 프리젠테이션 문서를 통하여 장치를 제어하고 모니터링 할 수 있다. 즉, Jini를 사용하여 인터넷의 사용자에게 댁내 장치 제어를 위한 인터페이스를 제공하기 위해서는 별도의 인터페이스를 동적으로 생성하여 제공하여야 하며, 이는 해당 장치에 최적화 된 것은 아니다. 반면 UPnP를 사용할 경우 해당 장치의 프리젠테이션 문서를 그대로 이용할 수 있기 때문에 사용자에게 최적의 제어 인터페이스를 제공할 수 있다.

2.2 인터넷에서 UPnP 기반 홈 네트워크 장치 제어 방안

외부 인터넷에서 홈 네트워크의 장치를 제어하기 위해서는 이 두 네트워크를 연결하는 홈 게이트웨이의 역할이 필수이다. 앞서 언급하였듯이 IPv4의 주소 공간의 부족과 보안의 이유로 홈 네트워크는 사설망으로 주로 구성되며 이는 인터넷과 홈 네트워크의 단절을 의미한다. 따라서 이 두 네트워크를 연결하기 위해서는 홈 게이트웨이가 반드시 필요하다. 이러한 홈 게이트웨이는 단순히 두 네트워크를 연결하는 역할 만을 수행하지 않고 앞서 언급한 미들웨어의 한계점을 극복할 수 있는 기능을 수행하여 사용자가 외부 인터넷에서 홈 네트워크의 장치를 제어할 수 있도록 한다. 이를 위한 방법으로는 홈 게이트웨이에 모든 기능이 집중된 방법과 홈 게이트웨이와 클라이언트에 기능을 분산한 방법으로 나눌 수 있으나, 이 두 방법 모두 홈 게이트웨이가 수행하여야 하는 역할은 유사하다. 홈 게이트웨이는 컨트롤 포인트의 역할을 수행하여 홈 네트워크에 접속한 장치를 발견하고 이들에 대한 정보를 유지하며 사용자에게 제어 가능한 장치의 리스트를 제공한다. 그리고 사용자가 제어할 장치를 선택하였을 때 해당 장치의 프리젠테이션을 전달한다. 이 때, 전달되는 프리젠테이션은 적절히 수정되어야 한다. 이는 사용자의 제어 명령이 홈 게이트웨이로 전송되도록 하여 홈 게이트웨이가 해당 장치를 제어할 수 있도록 하기 위함이다. 그리고 홈 네트워크에서 발생하는 다양한 이벤트를 사용자에게 능동적으로 알려줄 수 있는 기능을 가져야 한다. 본 논문에서 제안하는 UPnP 프록시 시스템은 후자의 방법을 사용한다.

• 홈 게이트웨이에 모든 기능이 집중된 장치제어

홈 게이트웨이에 원격 제어에 필요한 모든 기능을 두어 사용자는 단순히 웹 브라우저만을 사용하여 홈 게이트웨이에 접속함으로써 댁내의 장치를 제어할 수 있다. 이 방법은 사용자에게 필요한 프로그램을 설치하는 번거로움을 주지 않는다. 그러나 클라이언트에는 브라우저만이 동작하기 때문에 장치의 이벤트를 실시간으로 전송받을 수 없다. 웹 브라우저는 단순한 요청/응답 방식으로 동작하며 데이타를 능동적으로 수신할 수 없기 때문에 브라우저에서 명시적인 데이타 요청을 하지 않는 한 데이타를 수신할 수 없다. 따라서 이러한 방식을 사용하기 위해서는 주기적인 장치의 상태 요청이 필수적이다. 이 외에 이벤트를 수신할 수 있는 모듈을 애플릿 또는 DLL 등으로 구현하여 동적으로 다운로드 받아 이벤트를 수신할 수 있으나, 클라이언트로 전송되어야 할 프로그램의 크기가 크고 보안과 기능상 문제점을 가진다. 그리고 원본 프리젠테이션은 장치의 제어를 위하여 UPnP API를 사용한다. 그러나 클라이언트에는 이를 위한 프로그램이 없기 때문에 홈 게이트웨이는 각각의 API에 대하여 해당 동작을 위한 메시지를 전송할 수 있도록 원본 프리젠테이션을 수정하여 클라이언트로 전송해야 한다.

• 클라이언트에 UPnP API를 제공하는 응용 프로그램을 설치하는 방법

이 방법은 홈 게이트웨이와 클라이언트에 기능을 분산하여 사용자에게 장치 제어 방법을 제공하는 방법이다. 클라이언트에 Common Object Model(COM) 기술을 이용한 UPnP API를 제공하여 홈 게이트웨이에 모든 기능이 집중된 장치제어 방법에서 나타난 문제점을

해결할 수 있다. 이 API는 기존에 윈도우 시스템에 설치되어 있는 API[4]와 그 형태와 사용방법은 같으나, 사용자가 이를 호출하였을 때 홈 게이트웨이로 작업 내용을 전달하는 역할을 수행한다. 그리고 원본 프리젠테이션이 기존의 윈도우에서 제공하는 API대신 새로 설치된 API를 사용하도록 수정하여 프리젠테이션 문서를 최소한의 수정으로 사용할 수 있다. 또한, 프리젠테이션 외에 API를 사용하는 일반 응용 프로그램을 통해서도 홈 네트워크 외부에서 장치를 제어할 수 있다. 또한 홈 네트워크에서 발생하는 장치의 이벤트 메시지를 받을 수 있는 모듈을 클라이언트에 설치하여 장치의 이벤트를 전송받아 장치의 상태를 실시간으로 사용자에게 보여줄 수 있다. 이벤트를 수신하였을 경우 사용자가 미리 등록한 콜 백 함수를 호출함으로써 사용자가 원하는 작업을 수행할 수 있다. 이 방법은 클라이언트에 응용 프로그램을 미리 설치해야 한다는 단점이 있다. 또한 기존의 윈도우 시스템에는 UPnP API[4]가 설치되어 있으므로, 새로운 API를 설치한 후 두 API를 모두 사용할 수 있는 방법 또한 필요하다. 즉, 홈 네트워크에서는 기존의 API를 사용하고 외부에서는 새로 설치된 API를 사용할 수 있어야 한다.

2.3 사례 연구

현재 가장 대표적인 UPnP 구현물은 Windows ME/XP에 내장된 UPnP 모듈[4]이다. 이 모듈은 사용자의 호스트에 컨트롤 포인트와 API를 제공하여 사용자가 인터넷 익스플로러와 프리젠테이션 문서를 이용하여 홈 네트워크 내부에서 장치를 발견하고 제어할 수 있다. 홈 네트워크에 장치가 연결되었을 때 사용자는 윈도우 탐색기를 통하여 장치의 연결을 실시간으로 알 수 있으며 이 장치를 선택함으로써 장치의 프리젠테이션 문서를 전송받아 장치를 제어할 수 있다. 그러나 이 시스템은 2장 1절에서 설명한 것처럼 단지 댁내에서만 장치를 발견하고 제어할 수 있다. 그러나 본 시스템은 사용자에게 위 시스템과 동일한 제어 환경을 인터넷에서도 사용할 수 있도록 한다. 즉, 사용자는 인터넷으로 댁내에 연결된 장치를 검색할 수 있으며, 인터넷 익스플로러와 프리젠테이션 문서를 이용하여 장치의 원격제어와 모니터링을 할 수 있다.

Prosyst사에서는 OSGi 게이트웨이에 Jini와 UPnP 디들웨어 장치 드라이버를 번들 형태로 구현하여 장착하였다[7]. 이는 Jini 장치와 UPnP 장치들을 OSGi라는 하나의 프레임워크를 통하여 관리하고 제어할 수 있는 방법을 제공한다. 그러나 이 시스템은 사용자의 원격 장치 제어 기능은 제공하지 않으며, OSGi를 통한 다양한 서비스의 제공과 UPnP, Jini 등의 미들웨어 간의 연동에 초점을 둔다. 그러나 본 시스템은 장치의 원격제어를 목적으로 하며, 이 하나의 목적을 위하여 장치 목록 생성, 프리젠테이션 자동 생성, HTTP 기반의 이벤트 전송 등의 기능을 제공한다.

일본의 Keio 대학에서는 Jini를 이용하여 홈 네트워크 외부에서 내부의 장치를 제어하고 실시간 스트리밍 데이타를 전송받을 수 있는 EXWeb 시스템[5]을 개발하였다. 사용자는 i-mode 휴대폰에 탑재된 클라이언트 프로그램을 이용하여 댁내의 홈 서버에 접속하고 홈 네트워크 내의 장치들을 검색할 수 있으며, 이들 장치 중 VCR을 제어하고 동영상과 음성을 전송받을 수 있다. 이 시스템의 핵심 모듈인 EXWeb 서버는 Jini의 룩 업 서비스를 제공하며 홈 네트워크와 인터넷을 연결한다. 그리고 장치가 제공하는 기능들을 기반으로 HTML 장치 제어 인터페이스를 자동으로 생성하여 사용자가 이를 통하여 장치를 제어할 수 있도록 한다. 그러나 자동으로 생성된 제어 인터페이스는 사용자에게 친숙도를 제공함에 있어 한계를 가진다. 즉, 제어 인터페이스는 장치가 제공하는 액션 리스트와 그 인자들을 기반으로 생성됨으로 인하여 제어 인터페이스 내의 명령이 사용자에게 익숙하지 않은 이름으로 표시될 수 있으며, 장치가 다양한 기능을 제공할 경우 인터페이스 화면의 구성에 있어 부자연스럽다. 그러나 본 시스템은 기존의 프리젠테이션 문서를 그대로 이용하기 때문에 사용자는 댁내에서와 동일한 제어 환경을 제공받을 수 있다.

3. UPnP 프록시 시스템의 설계와 구현

3.1 시스템의 구성

UPnP 프록시 시스템은 사용자가 브라우저(웹 브라우저 또는 Wireless Application Protocol(WAP) 브라우저)만을 이용하여 댁내의 가전 기기들을 원격 제어 할 수 있는 기능을 제공하며 그림 4와 같이 구성되어 있다.

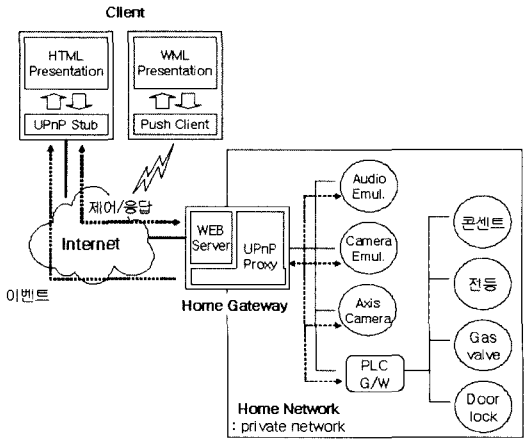

그림 4 UPnP Proxy 시스템의 구성도

홈 게이트웨이의 UPnP 프록시는 웹 서버와 연동하여 클라이언트의 원격 제어 기능을 제공하기 위하여 다양한 서비스를 제공한다. 홈 네트워크에 접속하고 해제하는 장치들을 발견하고, 이들의 정보를 이용하여 장치 리스트 웹 문서를 생성한다. 클라이언트로부터 전송된 사용자의 제어 명령에 따라 장치들을 직접 제어하고 이에 대한 응답 메시지를 전송하며, 홈 네트워크 내에서 장치의 이벤트가 발생할 경우 이를 HTTP 기반의 푸시 기술을 이용하여 클라이언트에게 능동적으로 이벤트 메시지를 전송한다. 이 외에 웹 문서간의 연결성과 기존의 UPnP API 대신 스터브를 사용할 수 있도록 하기 위하여 프리젠테이션을 적절히 수정하며, 프리젠테이션을 제공하지 않는 장치를 위하여 장치 디스크립션과 서비스 디스크립션을 기반으로 HTML 및 WML 프리젠테이션을 자동 생성한다.

유선 환경의 클라이언트에서는 HTML 브라우저(인터넷 익스플로러)와 UPnP 스터브의 상호 연동으로 장치 제어 인터페이스를 제공한다. 브라우저는 프리젠테이션 문서를 전송받은 후 자동으로 스터브를 실행한다. 스터브는 기존 윈도우에서 제공하는 API와 동일한 형태와 사용방법의 API를 제공함으로써 장치가 제공하는 프리젠테이션과 일반 응용 프로그램을 최소한의 수정으로 사용할 수 있도록 한다. 장치의 상태는 사용자의 제어 또는 주변 환경의 변화로 변화될 수 있으며, 프록시는 이러한 이벤트를 감지하여 이를 스터브로 전송한다. 이벤트 메시지를 받은 스터브는 프리젠테이션에서 설정한 이벤트 핸들러를 호출하여 사용자가 원하는 작업을 수행할 수 있도록 한다. 무선 환경의 클라이언트에는 스터브의 이벤트 수신 모듈만을 가지는 푸시 클라이언트가 WAP 브라우저와 연동하여 사용자에게 제어 인터페이스를 제공한다.

3.2 시스템 구조

UPnP 프록시 시스템은 그림 5와 같이 크게 클라이언트와 프록시 모듈로 구성된다. 클라이언트는 유선망 클라이언트를 위한 스터브와 무선망 클라이언트를 위한 푸시 클라이언트로, 프록시는 에이전트와 브리지로 구성된다. 그리고 이들 간의 통신은 원격제어 프로토콜[8]을 사용하여 이루어진다. 원격제어 프로토콜에 대해서는 다음 절에서 설명한다.

스터브는 [4]에서 정의된 것과 같이 다수의 인터페이스와 각 인터페이스에 다수의 메쏘드로 구성되어 있다. 사용자는 이러한 메쏘드, 즉, API를 이용하여 장치의 정보 획득, 제어, 이벤트 핸들러 등록 등의 작업을 수행할 수 있다. 최초 스터브가 생성될 때 스터브는 해당 장치의 장치 디스크립션과 서비스 디스크립션을 전송받아 장치의 기본 정보를 장치 정보 데이타베이스에 저장하여 현재의 상태 정보 이외의 기본적인 정보를 제공한다. 그리고 사용자가 장치 제어 등을 위하여 생성한 인터페이스는 메모리 누수현상을 막기 위하여 인터페이스 관리 모듈에서 관리한다. 이벤트 메시지 처리 모듈은 프록

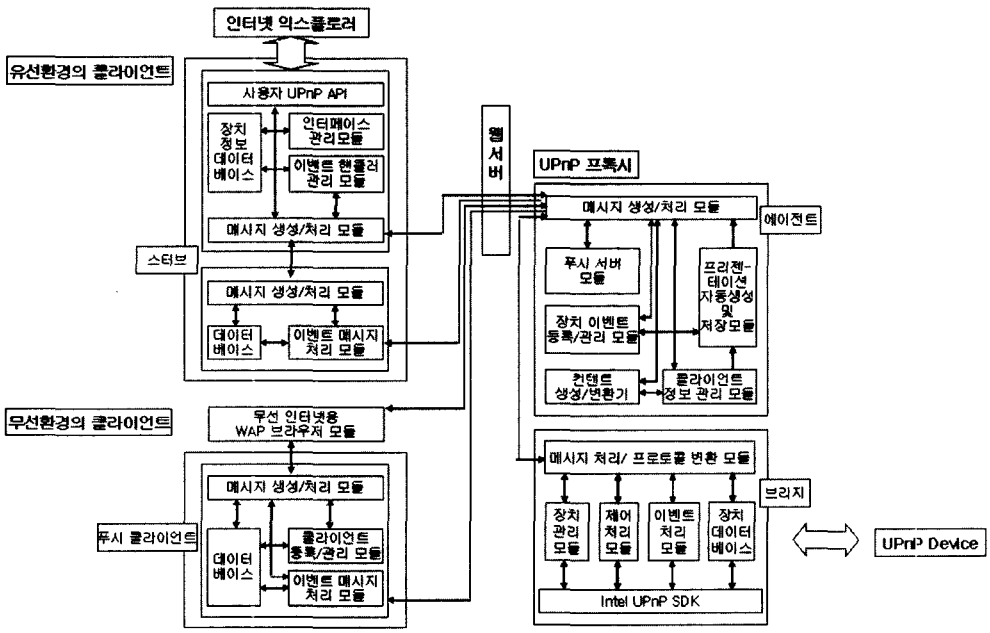

그림 5 UPnP Proxy 시스템의 모듈 구성도

시로부터 전송된 이벤트 메시지를 수신하며 이를 이벤트 핸들러 관리모듈로 전송한다. 이벤트 핸들러 관리 모듈은 사용자가 등록한 이벤트 핸들러의 주소를 가지고 있으며 전송된 이벤트에 따라 적절한 핸들러를 호출한다. 스터브 모듈은 HTML 문서와 일반 응용 프로그램 모두에게서 사용될 수 있으며, HTML 프리젠테이션에서는 비주얼 베이직 스트립트를 사용하여 이 모듈의 API를 호출한다.

푸시 클라이언트는 스터브의 이벤트 수신 모듈에서 다수의 브라우저를 지원하기 위한 모듈이 추가된 것이다. 프록시로부터 이벤트 메시지를 수신하며, 수신한 이벤트 메시지를 자신에게 접속한 적절한 브라우저에게 전송한다.

프록시는 브리지와 에이전트로 구성되며 이들 간의 연동을 통하여 사용자에게 원격 제어 기능을 제공한다. 브리지는 홈 네트워크의 장치들을 제어하고 관리하며, 에이전트는 사용자에게 컨텐츠의 생성, 변환, 전송 및 이벤트 전송 등을 수행한다.

에이전트는 다시 CGI 모듈과 데몬 모듈로 구분된다. CGI 모듈은 사용자에게 웹 문서를 전달하고, 사용자가 전달하는 제어 명령을 데몬 모듈로 전달하는 역할을 수행한다. 또한, UPnP 프록시와 웹 서버간의 연동을 위한 인터페이스 역할을 수행한다. 에이전트 데몬 모듈은 사용자의 브라우저 정보, 이벤트 등록 현황 등의 정보를 가지고 있어, 사용자의 환경에 적절한 웹 문서를 전달하고 이벤트가 발생하였을 경우 이 이벤트 메시지를 전달한다. 또한, 프리젠테이션 자동 생성 등의 사용자의 인터넷에서의 댁내 가전 제어를 위한 다양한 부가 기능을 제공한다.

브리지는 Intel의 UPnP SDK[9]를 사용하여 홈 네트워크에 접속한 장치들을 발견하고 관리한다. 장치 제어 모듈은 홈 네트워크에 접속하고 접속 해제하는 장치들을 발견하고 이들 장치들에 대한 정보를 장치 데이타베이스에 저장한다. 그리고 제어 처리 모듈은 사용자의 제어 명령에 따라 장치를 제어하고 그 응답 메시지를 전송하며 예외 상황이 발생할 경우 이를 처리한다. 이벤트 처리 모듈은 장치의 상태가 변하여 이벤트가 발생할 경우 이를 처리하는 모듈이다. 브리지를 기준으로 브리지와 장치 간에는 UPnP 포럼에서 정의된 메시지가 사용되며 브리지와 에이전트, 에이전트와 클라이언트 간에는 원격 제어 프로토콜이 사용된다. 따라서 브리지는 이들 두 프로토콜 간의 변환 작업을 수행하며, 이는 프로토콜 변환 모듈에서 수행된다.

UPnP 스펙에서는 장치가 프리젠테이션 문서를 제공하여야 하는가를 장치 제공 업자의 선택사항으로 정의하고 있다. 따라서 장치는 장치 제어를 위한 프리젠테이션 문서를 제공하지 않을 수 있다. 그러나 본 시스템은 사용자 제어 인터페이스로 장치의 프리젠테이션 문서를 사용하므로, 프리젠테이션 문서가 존재하지 않는 장치에 대해서는 자동으로 이를 생성할 수 있는 기능이 있어야 한다. 에이전트의 프리젠테이션 자동생성 및 저장 모듈에서는 장치의 장치 디스크립션과 서비스 디스크립션을 이용하여 유선 환경의 클라이언트를 위한 HTML 문서와 무선 환경의 클라이언트를 위한 WML 문서를 동적으로 생성한다. 그리고 HTML 프리젠테이션 문서에는 윈도우가 제공하는 API를 호출하도록 되어 있으므로 컨텐츠 생성/변환기는 이를 UPnP 스터브의 API를 호출하도록 프리젠테이션 문서를 수정한다. 또한 모든 웹 문서들 간의 연결성 유지를 위하여 웹 문서를 적절히 수정한다. 푸시 서버 모듈은 HTTP 푸시 기술을 기반으로 이벤트를 클라이언트로 전송한다.

3.3 원격 제어 프로토콜[8]의 정의 및 시스템 동작 흐름

클라이언트에서 홈 게이트웨이로 전송하는 메시지에는 제어하고자 하는 장치 명과 서비스 명, 그리고 다양한 정보들을 포함하고 있어야 한다. SOAP과 GENA 프로토콜에는 이러한 필요한 정보를 모두 표현할 수 있는 기능이 없으므로 이들 프로토콜을 수정 또는 확장하여 사용하여야 하지만, 이 프로토콜들은 클라이언트와 프록시 간에 불필요한 정보를 많이 포함하고 있으므로 UPnP 프록시 시스템에서는 원격 제어 프로토콜을 정의하여 본 시스템을 최적화하였다.

원격 제어 프로토콜은 크게 세 가지로 분류된다. 정보 획득을 위한 메시지, 장치 제어 및 이벤트를 위한 메시지, 그리고 부가 서비스를 위한 메시지로 분류된다. 그리고 이 프로토콜은 기존의 GENA와 SOAP에서 부족한 정보를 보완하고 장치 제어를 위한 최소한의 정보만을 전송하도록 XML을 이용하여 디자인 되었다. 메시지 내부는 메시지 자체를 정의하는 타입 부분과 타입에 따른 정보를 기술하는 바디로 구성된다.

장치 리스트 갱신 메시지는 시스템이 부트 업 될 때와 홈 네트워크에 장치가 새로 접속하거나 접속 해제하였을 때 브리지 모듈이 에이전트 모듈에게 이벤트로 전송한다. 에이전트 모듈은 이 메시지를 기반으로 사용자에게 현재 접속한 장치들의 리스트를 웹 페이지 형식으로 보여주며 사용자가 장치를 선택하였을 때 해당 장치의 프리젠테이션 URL로부터 프리젠테이션 문서를 클라이언트에게 전달한다.

장치의 제어를 위한 메시지에는 제어하고자 하는 장치와 서비스의 ID와 함께 수행하고자 하는 액션과 그 파라미터들이 있다. 장치의 특정 동작을 의미하는 액션은 장치의 서비스에 속하고 한 장치에 같은 이름의 액

션이 다른 서비스에 여러 개가 존재할 수 있기 때문에 메시지 내에 반드시 장치의 Unique Device Name (UDN)과 서비스 ID를 기술하여야 한다. 그리고 제어 요청 메시지의 인자들은 입력 인자들이며 응답 메시지의 인자는 출력 인자들이다. 이 메시지는 클라이언트가 UPnP API인 IUPnPService::InvokeAction에 의하여 전송된다.

홈 네트워크 내의 장치의 상태 변화로 인하여 이벤트가 발생하였을 때 클라이언트가 이를 푸시 메시지로 전송받기 위하여 반드시 이벤트 등록을 하여야 한다. 클라이언트는 이벤트 등록을 하지 않는다면 장치를 제어할 수 있으나 그 장치의 상태를 실시간으로 모니터링 할 수 없다. 이를 위하여 프로토콜 메시지에는 이벤트 등록을 원하는 장치와 서비스의 ID가 포함된다. 장치는 장치 제어의 결과로써 변화된 장치의 상태를 제어 응답 메시지로써 클라이언트에게 전송할 수 있으나 이벤트로써도 전송할 수 있다. 이는 장치에 따라 다르다. 그리고 다른 사용자에 의하여 또는 다른 장치간의 상호 동작에 의하여 장치의 상태가 변화될 수 있으므로 장치의 상태를 실시간으로 모니터링 하기 위해서는 반드시 이벤트 등록을 하여야 한다. 그리고 더 이상 해당 장치에 대하여 메시지를 받지 않기를 원한다면 이벤트 등록 해제 메시지를 전송한다. 이러한 과정은 IUPnPService::AddCallback에 의하여 이루어진다. UPnP 스터브는 이 API가 호출되었을 때 클라이언트가 지정한 콜 백 함수의 위치를 저장하고 프록시에 이벤트 등록 메시지를 전송한다.

이 외에도 장치의 상태 정보를 질의하는 메시지, 오류가 발생하였을 때의 메시지 등 다양한 메시지가 있으며 클라이언트는 이러한 메시지를 UPnP 프록시로 전송함으로써 다양한 서비스를 제공받을 수 있다. 그리고 각각의 메시지는 각 UPnP API와 매핑된다.

그림 6은 전체 시스템이 부트 업 된 후, 사용자가 자신이 제어할 장치를 선택하였을 때 이루어지는 메시지의 흐름을 보인다. 이는 사용자가 장치에 대한 제어 명령을 하기 전 준비단계에 해당한다. 여기에는 보안을 위해서 SSL 연결을 설정하고 ID와 암호를 이용하여 인증 작업을 수행하는 과정은 생략되고 장치를 제어하기 위한 과정만을 나타내었다. 그리고 스터브와 에이전트, 에이전트와 브리지 간의 통신에는 HTTP 메시지와 원격 제어 프로토콜 메시지가 사용되며 브리지와 장치 간에는 HTTP와 UPnP 프로토콜이 사용된다.

1. 시스템이 부트업되면 브리지는 SSDP와 GENA를 이용하여 홈 네트워크에 연결된 장치를 발견하며 장치 디스크립션 문서를 모두 전송받는다.
2. 에이전트는 브리지에게 현재 연결된 장치의 리스트를

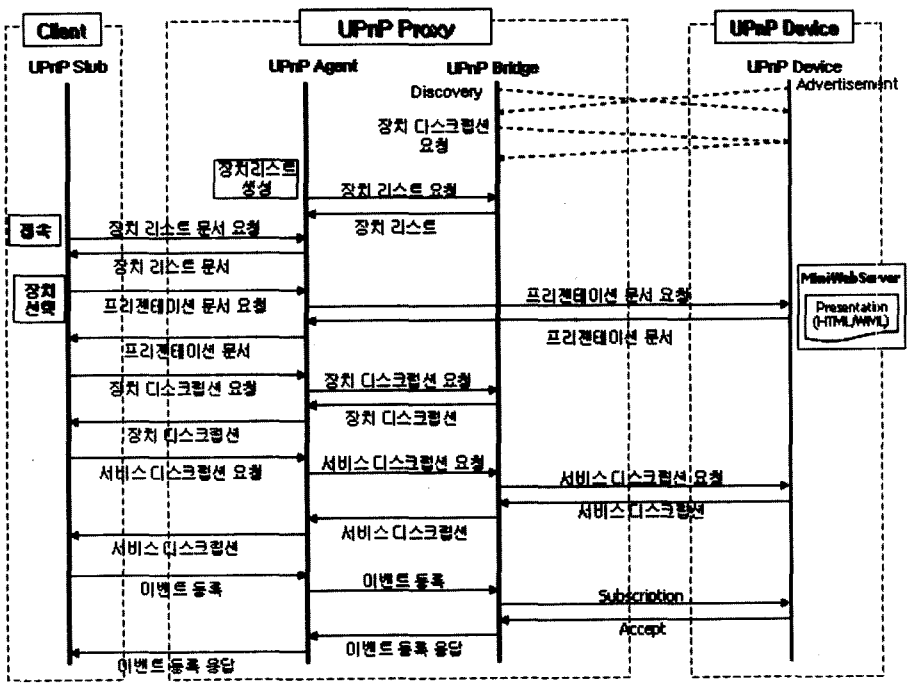

그림 6 시스템이 부트업 될 때부터 사용자가 장치를 선택한 후까지의 메시지 흐름

요청하며, 이 리스트를 이용하여 장치 리스트 웹 문서를 생성한다.

3. 사용자는 홈 게이트웨이의 URL로 접속하여 장치 리스트 웹 문서를 전송받으며 이 문서를 통하여 자신이 제어할 장치를 선택한다.
4. 장치가 선택되면 에이전트는 장치로부터 프리젠테이션 문서를 전송받아 적절히 수정한 후 이를 클라이언트에게 전송한다. 만약 프리젠테이션 문서가 없다면 자동으로 프리젠테이션 문서를 생성하여 이를 전송한다.
5. 클라이언트(웹 브라우저)는 프리젠테이션 문서 내의 스크립트를 수행 시작하며 이 과정에서 장치 디스크립션과 서비스 디스크립션이 전송된다. 이 과정으로 UPnP 스터브는 현재 상태 값을 제외한 기본 장치의 정보를 가지게 된다.
6. 마지막으로 사용자가 제어할 장치에 대하여 이벤트 등록을 함으로써 해당 장치에 이벤트가 발생하였을 경우 이벤트 메시지를 수신할 수 있도록 한다.

그림 7은 사용자가 프리젠테이션 문서를 이용하여 장치를 제어할 때의 메시지 흐름이다.

1. 사용자는 프리젠테이션 문서의 버튼 등을 통하여 장치 제어 명령을 내린다.
2. 프리젠테이션 문서 내의 스터브의 API 함수가 호출되며 스터브는 에이전트에게 장치 제어 명령을 전송한다.
3. 에이전트는 이 메시지를 브리지에게 전송하며 브리지는 이 메시지를 SOAP 메시지로 변환하여 장치로 전송한다.
4. 장치로부터 전송된 제어 응답 메시지를 클라이언트에게 전송한다.
5. 장치에서 이벤트가 발생하였을 경우 브리지는 이를 에이전트로 전송하며, 에이전트는 푸시 기술을 사용하여 이를 클라이언트로 전송한다.

그림 8은 사용자가 모든 장치 제어 명령을 마치고 웹 브라우저를 종료하거나 다른 웹 사이트로 이동하였을 경우 UPnP 스터브가 종료하며 이루어지는 메시지 흐름이다.

1. 클라이언트는 에이전트에게 해당 장치에 대한 이벤트 등록 해제 메시지를 전송한다.
2. 에이전트는 이 메시지를 브리지에게 전송하며, 브리지는 장치로 이벤트 등록 해제 메시지를 전송한다.
3. 이에 대한 응답 메시지는 클라이언트로 전송되며 스터브는 종료한다.

3.4 무선 인터넷 환경의 클라이언트 지원

일반적으로 무선 환경의 클라이언트는 유선 환경의 데스크톱 컴퓨터에 비하여 네트워크의 대역폭이 낮고 프로세싱 능력이 낮으며 화면의 크기가 작다. 이러한 환경의 클라이언트를 위하여 UPnP 프록시 시스템은 사용자 제어 인터페이스로 HTML 프리젠테이션 문서 대신

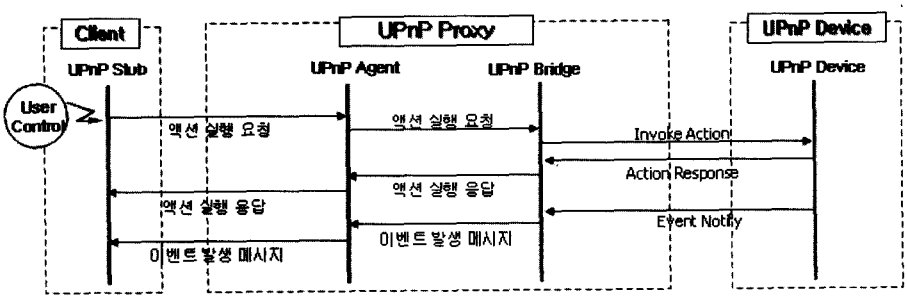

그림 7 장치 제어와 이벤트 전송을 위한 메시지 흐름

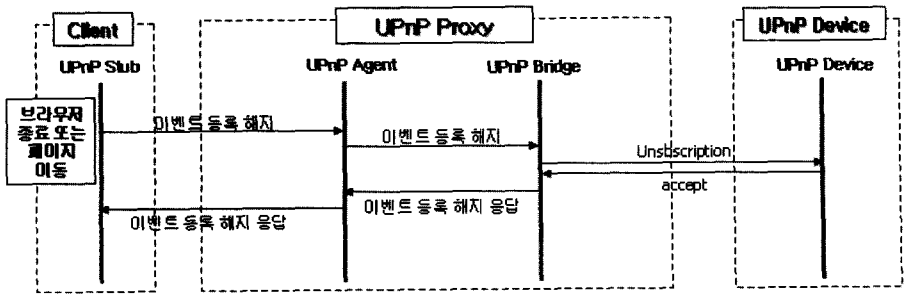

그림 8 장치 제어 종료를 위한 메시지 흐름

WML 프리젠테이션 문서를 제공하여 사용자에게 무선 인터넷에 최적화된 환경을 제공한다. UPnP 프록시 시스템에서 사용자는 이 WML 프리젠테이션 문서와 WAP 브라우저를 이용하여 장치를 제어할 수 있다.

클라이언트에는 푸시 클라이언트와 푸시 메시지를 수신할 수 있는 WML 브라우저가 탑재되어있다. 푸시 클라이언트는 프록시로부터 전송되는 이벤트 메시지인 푸시 메시지를 전송받아 브라우저에게 전달하고 브라우저로부터 전송되는 사용자의 입력 메시지를 원격 제어 프로토콜로 바꾸어 프록시에게 전송하는 역할을 수행한다.

표 1은 WML 프리젠테이션 문서 내의 장치 제어 부분이다. 사용자의 장치 제어 입력은 화면상에 표시되는 제어 명령을 선택함으로써 이루어지며, 이를 통하여 푸시 클라이언트에게 포스트 형식의 HTTP 메시지가 전송된다. 전송되는 메시지의 내용은 "UpnpDigitalCameraViewer1-2-000000000012.digital_camera_photo.Snapshot()"이다. 이는 "DeviceUDN.ServiceID.ActionName()"의 형식으로, 푸시 클라이언트가 이를 이용하여 앞서 설명한 원격제어 프로토콜 메시지로 변환하며, 이를 위하여 푸시 클라이언트는 장치의 기본적인 정보를 가지고 있어야 한다. 본 시스템이 정의하는 프리젠테이션 디스크립션은 WML 프리젠테이션을 사용하기 위한 최소한의 정보이며, 푸시 클라이언트는 이를 이용하여 원격 제어 프로토콜 메시지를 생성한다.

프리젠테이션 디스크립션은 장치의 UDN과 서비스 ID, 그리고 WML 프리젠테이션의 변수와 장치의 상태 변수의 관계를 기술한다. 표 1은 디지털 카메라의 프리젠테이션 디스크립션 문서로 <p_var> 태그의 변수는 프리젠테이션에서 사용되는 변수의 이름이고 <s_var> 태그의 변수가 실제 장치의 상태 변수이다. 이 프리젠테이션 디스크립션 문서를 통하여 UPnP 프록시는 WML 프리젠테이션 문서를 적절히 변형하여 전달하고 푸시 클라이언트는 WML 브라우저와 프록시간의 통신을 원활히 한다. 이 문서에 기술된 내용은 장치의 장치 디스크립션과 서비스 디스크립션에 기술된 모든 내용을 기술하지 않고 프리젠테이션에서 필요한 부분만을 가지고 있다. 무선 환경의 클라이언트를 위한 이 두 문서는 장치가 가지고 있을 수 있으며 또한 프록시가 장치 디스크립션과 서비스 디스크립션을 이용하여 동적으로 생성할 수 있다.

무선 환경에서 사용자가 장치를 제어하는 메시지 흐름은 3.3절에서 설명하는 것과 거의 동일하다. 무선 환경의 클라이언트와 유선 환경의 클라이언트는 사용자에게 제공하는 기능이 동일하기 때문이다. 메시지 흐름에서 다른 점은 그림 6에서 무선 환경의 클라이언트가 장치 디스크립션과 서비스 디스크립션을 클라이언트가 전송받는 대신, 클라이언트는 프리젠테이션 디스크립션을 전송받는다.

3.5 구현 환경 및 실험 결과

UPnP 프록시 시스템의 프록시는 리눅스를 기반으로 개발하여 이를 PowerPC 계열의 셋탑 박스 형태의 홈 게이트웨이에서 동작하도록 하였다. 클라이언트 부분인 스터브와 푸시 클라이언트 그리고 WAP 브라우저는 윈도우 또는 윈도우CE 기반으로 개발되었다. 그리고 테스트베드의 구축과 실험을 위하여 UPnP 장치와 동일한 동작을 하는 샘플 장치를 개발하였다. 오디오 에뮬레이터는 일반 PC상에서 동작하며, 카메라 에뮬레이터는 임베디드 보드와 일반 디지털 카메라를 연결하여 동작하도록 하였다. Axis 카메라는 해당 장치만으로도 UPnP 장치 기능을 수행하도록 개발된 웹 카메라이다. 본 연구에서는 유선 환경의 클라이언트와 무선 환경의 PDA는 공인 IP를 사용하는 인터넷으로 연결되어 있으며, 홈 네트워크는 사설 IP를 사용하여 홈 네트워크 내부와 외부가 분리되는 환경을 구축하였다. 그리고 홈 네트워크 외부에서의 제어가 댁내에서의 제어 결과가 동시에 동일한가를 테스트하기 위하여 사설망 내부에 기존의 Windows ME 시스템을 이용하여 외부 클라이언트와 동시에 직접 장치를 제어하였다. 구축된 테스트베드는 그림 9와 같으며, 개발 환경 및 동작 플랫폼은 표 2와 같다.

표 1 WML 프리젠테이션에서의 장치 제어의 예

```
<anchor>SnapShot
<go href="http://localhost:6000?
        agent=http://homegate.knu.ac.kr/cgi-bin/upnpproxy/agent_cgi" method="post">
<postfield name="ActionQuery"
  value="UpnpDigitalCameraViewer1-2-000000000012.digital_camera_photo.Snapshot()"/>
</go>
</anchor>
<br/>
```

표 2 UPnP 프록시 시스템 구현 환경

주요 모듈	동작 플랫폼	구현 환경 및 사용 언어
프록시	PowerPC series 82xx Linux Kernel 2.2.x	Intel Pentium 3, Linux Kernel 2.2.4 MontaVista Linux Professional Edition 2.1
스터브	Intel Pentium 3 Windows 2000/ME/XP	Intel Pentium 3 Visual C++ 6.0
푸시 클라이언트	iPAQ Pocket PC WinCE	Intel Pentium 3 Microsoft Embedded Visual Tools 3.0
WAP 브라우저	iPAQ Pocket PC WinCE	Intel Pentium 3 Microsoft Embedded Visual Tools 3.0
오디오 에뮬레이터	Intel Pentium 3 Linux Kernel 2.4.x	Intel Pentium 3, Kernel 2.2.4 GNU gcc 2.96 Intel UPnP SDK 1.0.4
디지털 카메라 에뮬레이터 및 PLC 장치	Hyper 104 evaluation board (strongARM) Linux Kernel 2.4.x Planet PLC Module	Intel Pentium 3, Kernel 2.2.4 arm-gcc cross-compiler 2.96 Intel UPnP SDK 1.0.4

표 3 HTML 프리젠테이션의 수정 전과 후

dim DSDocu **set DSDocu =** **CreateObject("UPnP.DescriptionDocument.1")** DSDocu.Load("device_description.xml") dim AudioDevice set AudioDevice = DSDocu.RootDevice dim ControlService set ControlService = AudioDevice.Services("...") ControlService.AddCallback GetRef("eventhandler") ... sub eventhandler(type, svcObj, varName, value) if(type = "VARIABLE_UPDATE") then case "PlayStatus" ... case "Volume" ...하략.	dim DSDocu **set DSDocu =** **CreateObject("KNU_UPnP.DescriptionDocument.1")** DSDocu.Load("device_description.xml") dim AudioDevice set AudioDevice = DSDocu.RootDevice dim ControlService set ControlService = AudioDevice.Services("...") ControlService.AddCallback GetRef("eventhandler") ... sub eventhandler(type, svcObj, varName, value) if(type = "VARIABLE_UPDATE") then case "PlayStatus" ... case "Volume" ...하략.

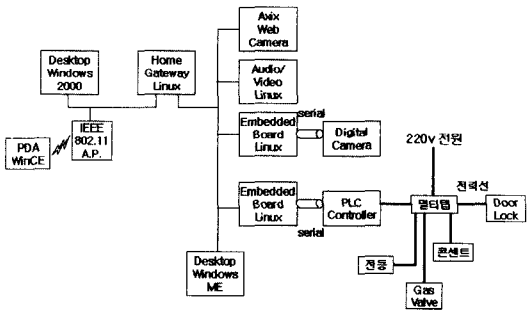

그림 9 UPnP Proxy System 테스트 베드 환경 구성

이러한 시스템을 이용하여 인터넷의 사용자는 홈 게이트웨이상의 UPnP 프록시에 접속하여 자신의 ID와 암호를 입력하면, 홈 네트워크에 접속한 카메라, 전등, 가스 밸브 등의 장치를 확인할 수 있다. 이 중, 하나의 장치를 사용자가 선택하게 되면 해당 장치를 위한 프리젠테이션 문서가 전송되어, 사용자는 이를 통하여 장치를 제어하고 모니터링 할 수 있다. 이러한 장치 제어 화면은 그림 10과 같다. 이렇게 UPnP 프록시 시스템을 이용하여 댁내 가전을 제어하고 모니터링 할 경우에는 많은 양의 데이타 교환이 필요하지 않기 때문에, 사용자가 느끼는 지연 시간은 댁내에서 장치를 제어하는 지연 시간과 큰 차이를 발견할 수 없었다. 하지만 대용량의 데이타 전송이 필요한 멀티미디어 서비스 등을 향후 제공할 경우, 많은 성능의 저하가 예상된다. 이러한 문제는 향후 멀티미디어 서비스를 제공함에 있어 반드시 고려되어야 할 것이다.

표 3은 장치에 존재하는 HTML 프리젠테이션을 UPnP 프록시를 경유하여 클라이언트로 전송될 경우 프록시가 이를 수정하기 전과 후의 프리젠테이션 내용이다. 수정

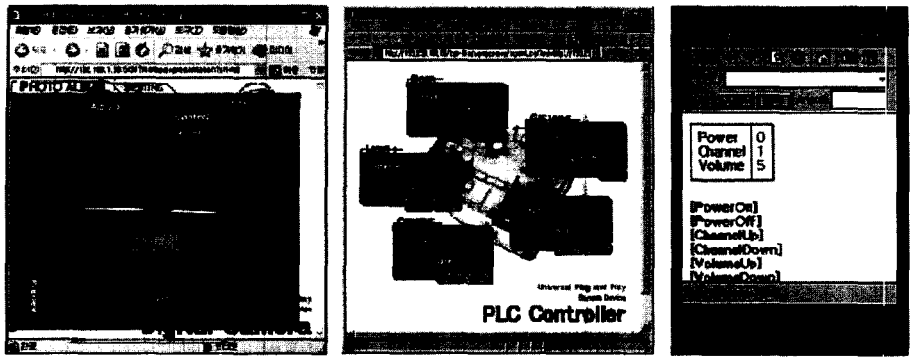

그림 10 디지털 카메라, PLC 장치, 그리고 오디오 에뮬레이터의 제어 화면

전의 문서와 수정 후의 문서와의 차이점은 프록시가 기존 UPnP API 객체를 생성하는 것을 스터브를 생성하도록 수정한 것이다. 즉, 수정 전 문서의 기존 윈도우용 UPnP API를 생성하는 "set DSDocu = CreateObject("UPnP.DescriptionDocument.1")"를 스터브를 생성하기 위하여 프록시가 "set DSDocu = CreateObject ("KNU_UPnP.DescriptionDocument.1")"로 수정한다. 이를 통하여 인터넷 브라우저는 스터브 객체를 생성하고 로딩하여 이 외의 VB 스크립트 코드는 그대로 사용하면서 원격제어를 가능하게 한다. 스터브는 COM 객체이기 때문에 VB 스크립트 뿐 아니라 일반 응용 프로그램에서도 사용될 수 있다. 그리고 표 3과 같이 스터브는 기존의 윈도우 운영체제에 포함된 UPnP API와 그 사용법이 동일하기 때문에 프리젠테이션 또는 일반 응용 프로그램을 스터브를 이용하여 새로이 작성하려 한다면 기존에 장치를 제어하는 방법과 동일한 방법으로 프로그램을 작성하여 장치를 제어할 수 있다는 장점을 가진다.

그림 10은 인터넷에서 인터넷 익스플로러와 PDA의 WAP 브라우저를 이용하여 카메라와 PLC 장치, 그리고 오디오 에뮬레이터를 동작시키는 그림이다. HTML 프리젠테이션은 기존의 윈도우용 UPnP API를 이용하여 작성한 것으로 댁내에서 장치를 제어하기 위한 일반적인 프리젠테이션 문서이다. 인터넷의 사용자가 이를 UPnP 프록시를 통하여 전송받게 되면, 클라이언트에서 기존의 UPnP API 모듈 대신 UPnP 스터브 모듈을 로딩하도록 프록시가 프리젠테이션을 수정하여 사용자는 인터넷에서도 기존의 프리젠테이션을 사용하여 장치를 제어할 수 있게 된다.

무선 환경의 사용자는 HTML 프리젠테이션에 비교하여 단순하지만 보다 작은 크기를 가지는 WML 프리젠테이션을 이용하여 장치를 제어한다. 특히 텍스트 형태의 WML 프리젠테이션을 프록시는 바이너리 형태로 변형하기 때문에 보다 전송되는 파일의 크기를 줄이기 때문에 무선 환경의 대역폭을 보다 효율적으로 사용한다. WML 문서는 HTML에 비교하여 사용자 UI를 구성하는 데에 많은 한계를 가지므로 비교적 단순한 UI를 제공한다.

그리고 하나의 장치에 대하여 여러 개의 프리젠테이션 문서를 통하여 동시에 제어할 경우 장치의 이벤트를 통하여 각 프리젠테이션의 장치의 상태가 동기화 되는 것을 확인할 수 있다. 이는 장치의 상태가 변경될 때 발생하는 GENA 메시지를 프록시가 전송받아 현재 해당 장치를 제어중인 클라이언트에게 푸시 메시지를 이용하여 장치의 이벤트를 전송하기 때문이다.

4. 결론 및 향후 연구 과제

현재 IP 기반의 홈 네트워크 제어 미들웨어는 Jini와 UPnP로 대표되며, 홈 네트워크 내에서의 장치 제어 방안을 제공하고 있다. 따라서 실제적인 홈 네트워크가 구축된다면 이들의 기능은 확장될 필요가 있으며 본 연구에서는 이를 위하여 UPnP 프록시 시스템을 구축하였다.

UPnP 프록시 시스템은 인터넷의 사용자가 댁내의 정보 가전을 모니터링하고 제어하기 위한 시스템으로 해당 장치가 제공하는 프리젠테이션을 이용한다. 이는 본 시스템의 가장 큰 장점으로서 사용자에게 댁 외에서도 댁내와 같은 제어 환경을 제공한다. 이는 클라이언트에 기존에 시스템이 사용하는 API와 형태와 사용 방법이 동일한 API를 제공하는 스터브를 설치함으로써 가능하다. 그리고 프리젠테이션을 제공하지 않는 장치를 위하여 프록시는 자동으로 프리젠테이션을 생성하여 제공한다. 또한, UPnP 프록시 시스템은 무선 환경의 사용자에게 최적의 환경을 제공한다. 무선 환경은 유선 환경에 비하여 대역폭의 차이와 더불어 사용자 단말기 또한 상대적으로 작다. 따라서 본 시스템은 기존의 HTML 프리젠테이션뿐만 아니라 무선 단말기를 위한 WML 프리

젠테이션을 제공하여 사용자에게 원격 제어 환경을 제공한다. 그리고, UPnP 프록시 시스템은 클라이언트에게 HTTP 기반의 푸시 기술을 사용하여 메시지를 전송할 수 있다. 이는 홈 네트워크에 이벤트가 발생하였을 경우 이를 실시간으로 사용자에게 알려줄 수 있는 기능을 제공한다. 이를 통하여 사용자는 홈 네트워크에서 특정 이벤트가 발생하였는가를 알 수 있으며, 특정 장치의 상태를 실시간으로 모니터링할 수 있다.

현재의 시스템은 장치의 모니터링과 제어 기능만을 제공하지만 홈 네트워크 장치의 큰 비중을 차지할 엔터테인먼트 장치를 위하여 멀티미디어 서비스를 제공하기 위한 방안이 필요하다. DHWG와 UPnP 포럼에서는 이러한 멀티미디어 또는 스트리밍 서비스를 위하여 AV 아키텍쳐 표준을 정의하였으며, 현재 구체적인 미디어 포맷과 그 서비스 제공 방안에 대하여 활발히 표준화가 진행중이다. 이러한 멀티미디어 서비스는 이를 위한 표준이 정의가 된 후에 이를 인터넷에서도 댁내의 서비스를 이용할 수 있는 방안에 대하여 연구가 되어야 할 것이다. 또한, 장치 제어에 있어 현재는 사용자가 UPnP 프록시에게 접속을 할 때에만 인증 과정을 수행하고 기존의 데이타 암호화 기법인 SSL을 이용하여 클라이언트인 스터브와 프록시 간의 데이타 보안을 유지하지만 사용자의 등급에 따라 장치를 제어할 수 있도록 사용자의 장치에 따른 권한 부여 및 보안 정책 수립 방안도 가져야 할 것이다. 최근 UPnP 포럼에서는 이러한 사용자 권한 부여 등을 위하여 Access control List(ACL)를 이용한 표준을 제정하였다[11]. 이러한 기능은 UPnP 프록시 시스템과는 분리된 기능으로써, 차후에 프록시 시스템과 연계되는 부분에 있어 새로이 연구가 진행되어야 할 것으로 보인다. 댁내에는 다양한 종류의 장치가 있을 것이며 각 장치에 대하여 특정 권한을 가진 사용자만이 제어를 할 수 있는 기능은 홈 네트워크에서 반드시 필요한 기능으로 생각된다.

참 고 문 헌

[1] Microsoft Corporation, "Universal Plug and Play Device Architecture," Jun. 2000, Available to: http://www.upnp.org

[2] Microsoft Corporation, "Understanding Universal Plug and Play," Jun. 2000, Available to: http://www.upnp.org

[3] Sun Microsystems, "Jini 1.1," 2000, Available to: http://developer.java.sun.com/developer/products/jini/

[4] Microsoft Corporation, "UPnP Development Kit Beta 2," Sep. 2000, Available to: http://www.microsoft.com/hwdev/upnp

[5] R. Yoshida, A. Inoue, J. Hiraishi, H. Shigeno, Y. Matsushita, "EXWeb: remotely operating devices in the home network," In Proceedings of 2002 IEEE 4th International Workshop on Networked Appliances, pp. 267-274, 2002.

[6] Dong-Sung Kim, Jae-Min Lee, Wook Hyun Kwon, In Kwan Yuh, "Design and implementation of home network systems using UPnP middleware for networked appliances," IEEE Transactions on Consumer Electronics, Vol. 48, Issue 4, pp. 963-972, Nov. 2003.

[7] P. Dobrev, D. Famolari, C. Kurzke, B. A. Miller, "Device and Service Discovery in Home Networks with OSGi," IEEE Communications Magazine, Vol. 40, Issue 8, Aug. 2002.

[8] Kyung Ho Chae, Dong Hee Kim, Kyungshik Lim, Jung Suk Park, "An XML-Based Protocol for the Remote Control of the UPnP Devices in the Internet," EALPIIT 2002, pp. 253-259, Jan. 2002.

[9] Linux SDK for UPnP Devices, Intel Corporation, 2003, Available to: http://upnp.sourceforge.net/

[10] UPnP Forum, "Universal Plug and Play Vendor's Implementation Guide," 2001, Available to: http://www.upnp.org

[11] UPnP Forum, "UPnP Security Ceremonies Design Document," 2003, Available to: http://www.upnp.org

김 동 희

2000년 2월 경북대학교 컴퓨터과학과 학사. 2002년 2월 경북대학교 컴퓨터과학과 석사. 2004년 2월 경북대학교 정보통신학과 박사수료. 2004년 1월~현재 한국전자통신연구원 홈네트워크그룹 연구원. 관심분야는 홈네트워크킹, 이동컴퓨팅, 유비쿼터스컴퓨팅

임 경 식

1982년 2월 경북대학교 전자공학과 공학사. 1985년 2월 한국과학기술원(KAIST) 전산학과 공학석사. 1994년 12월 University of Florida, 전산학과 공학박사. 1985년 2월~1998년 2월 한국전자통신연구원 책임연구원, 실장. 1998년 3월~현재 경북대학교 컴퓨터과학과 부교수

이 화 영

2002년 2월 경북대학교 컴퓨터과학과 학사. 2004년 2월 경북대학교 컴퓨터과학과 석사. 2004년 1월~현재 LG전자 솔루션기술그룹 연구원. 관심분야는 홈네트워크, 멀티미디어네트워크, 그리드컴퓨팅, 유비쿼터스컴퓨팅

안 준 철

2002년 2월 경북대학교 컴퓨터과학과 학사. 2004년 2월 경북대학교 컴퓨터과학과 석사. 2004년 1월~현재 유시스 디바이스지원팀. 관심분야는 홈네트워크킹, 이동컴퓨팅

조 충 래

1994년 2월 부산대학교 전자계산학과 전산학 학사. 1996년 2월 부산대학교 전자계산학과 전산학 석사. 1996년 3월~2000년 7월 한국기계연구원 연구원. 2000년 7월~현재 한국전자통신연구원 홈네트워크그룹 선임연구원. 관심분야는 홈네트워킹, 컴퓨터통신, 정보검색, Pervasive 컴퓨팅

박 광 로

1982년 경북대학교 전자공학과(학사). 1985년 경북대학교 대학원(석사). 2002년 충북대학교 대학원(박사). 1984년~현재 한국전자통신연구원 디지털홈연구단 홈네트워크그룹장(책임연구원). 관심분야는 홈서버/홈게이트웨이 기술, UWB 기술, 홈네트워크 미들웨어기술, 상황인지 센서기술